Interactive Shape Modeling and Deformation

T3: Half-Day Tutorial

Introduction, organization
Speakers

- **Olga Sorkine**
  Media Research Lab, VLG
  Courant Institute, New York University

- **Mario Botsch**
  Graphics & Geometry Group
  Bielefeld University
Shapes and Deformations

- Manually modeled and scanned shape data
- Continuous and discrete shape representations
Shapes and Deformations

- Why deformations?
  - Sculpting, customization
  - Character posing, animation

- Criteria?
  - Intuitive behavior and interface
  - Interactivity
Tutorial Goals

- Present recent research in shape editing
- Discuss practical considerations
  - Flexibility
  - Numerical issues
  - Admissible interfaces
- Comparison, tradeoffs
Schedule

09:00 – 09:10   Intro (O)
09:10 – 09:25   Shape representations, differential geometry primer (O)
09:25 – 10:05   Linear surface-based deformations (M)
10:05 – 10:30   Linear space deformations (O)
10:30 – 11:00   Break
<table>
<thead>
<tr>
<th>Time</th>
<th>Session Title</th>
<th>Instructor(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>11:00 – 11:10</td>
<td>Summary of linear methods</td>
<td>(M)</td>
</tr>
<tr>
<td>11:10 – 11:40</td>
<td>Nonlinear surface-based deformations</td>
<td>(M)</td>
</tr>
<tr>
<td>11:40 – 12:20</td>
<td>Nonlinear space deformations</td>
<td>(O)</td>
</tr>
<tr>
<td>12:20 – 12:30</td>
<td>Wrap-up</td>
<td>(O+M)</td>
</tr>
</tbody>
</table>
Interactive Shape Modeling and Deformation

T3: Half-Day Tutorial

Shape Representations
Differential Geometry Recap
Continuous/analytical surfaces

- Tensor product surfaces (e.g. NURBS)
- Subdivision surfaces

“Editability” is inherent to the representation
Spline Surfaces

- Tensor product surfaces ("curves of curves")
  - Rectangular grid of control points

\[ p(u, v) = \sum_{i=0}^{k} \sum_{j=0}^{l} p_{ij} N_i^n(u) N_j^n(v) \]
Spline Surfaces

- Tensor product surfaces ("curves of curves")
  - Rectangular grid of control points
  - Rectangular surface patch
Spline Surfaces

- Tensor product surfaces ("curves of curves")
  - Rectangular grid of control points
  - Rectangular surface patch

- Problems:
  - Many patches for complex models
  - Smoothness across patch boundaries
  - Trimming for non-rectangular patches
Subdivision Surfaces

- Generalization of spline curves / surfaces
  - Arbitrary control meshes
  - Successive refinement (subdivision)
  - Converges to smooth limit surface
  - Connection between splines and meshes
Subdivision Surfaces

- Generalization of spline curves / surfaces
  - Arbitrary control meshes
  - Successive refinement (subdivision)
  - Converges to smooth limit surface
  - Connection between splines and meshes
Spline & Subdivision Surfaces

- Basis functions are smooth bumps
  - Fixed support
  - Fixed control grid
- Bound to control points
  - Initial patch layout is crucial
  - Requires experts!
- Decouple deformation from surface representation!

Olga Sorkine, Courant Institute 3/30/2009
Discrete Surfaces: Point Sets, Meshes

- Flexible
- Suitable for highly detailed scanned data
- No analytic surface
- No inherent “editability”

Mesh editing
Differential Geometry

- Tool to analyze shape
- Key notions:
  - Tangents and normals
  - Curvatures
  - Laplace-Beltrami operator
Continuous Case – Parametric

- Local parameterization:
  \[
  \mathbf{p}(u,v) = \begin{pmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{pmatrix}, \quad (u,v) \in D \subset \mathbb{R}^2
  \]

- Tangent plane at point \( \mathbf{p}(u,v) \) is spanned by
  \[
  \mathbf{p}_u = \frac{\partial \mathbf{p}(u,v)}{\partial u}, \quad \mathbf{p}_v = \frac{\partial \mathbf{p}(u,v)}{\partial v}
  \]

- Normal: \( \mathbf{n}(u,v) = \frac{\mathbf{p}_u \times \mathbf{p}_v}{\| \mathbf{p}_u \times \mathbf{p}_v \|} \)
Discrete Case – Piecewise Linear

- No derivatives!
- Strategy 1: locally fit an analytic patch
  - Expensive
- Strategy 2: generalize definitions to discrete case
  - Fast
  - Start from intrinsic notions (non-parametric)
Normal Curvature

\[ \mathbf{n} = \frac{\mathbf{p}_u \times \mathbf{p}_v}{\| \mathbf{p}_u \times \mathbf{p}_v \|} \]

Direction \( \mathbf{t} \) in the tangent plane:

\[ \mathbf{t} = \cos \varphi \frac{\mathbf{p}_u}{\| \mathbf{p}_u \|} + \sin \varphi \frac{\mathbf{p}_v}{\| \mathbf{p}_v \|} \]
The curve $\gamma$ is the intersection of the surface with the plane through $\mathbf{n}$ and $\mathbf{t}$.

Normal curvature:

$$\kappa_n(\varphi) = \kappa(\gamma(p))$$

Curvature on a curve: the rate of change in normal
Normal Curvature

The curve $\gamma$ is the intersection of the surface with the plane through $n$ and $t$.

Normal curvature:

$$\kappa_n(\varphi) = \kappa(\gamma(p))$$

Curvature on a curve: the rate of change in normal
The curve $\gamma$ is the intersection of the surface with the plane through $\mathbf{n}$ and $\mathbf{t}$.

Normal curvature:

$$\kappa_n(\varphi) = \kappa(\gamma(p))$$

Curvature on a curve: the rate of change in normal
Normal Curvature

The curve $\gamma$ is the intersection of the surface with the plane through $n$ and $t$.

Normal curvature:

$$\kappa_n(\varphi) = \kappa(\gamma(p))$$

Curvature on a curve: the rate of change in normal
The curve $\gamma$ is the intersection of the surface with the plane through $n$ and $t$.

Normal curvature:

$$\kappa_n(\varphi) = \kappa(\gamma(p))$$

Discrete curvature: turning angles
Surface Curvatures

- **Principal curvatures**
  - Maximal curvature \( \kappa_1 = \kappa_{\text{max}} = \max_{\phi} \kappa_n(\phi) \)
  - Minimal curvature \( \kappa_2 = \kappa_{\text{min}} = \min_{\phi} \kappa_n(\phi) \)

- **Mean curvature**
  \[
  H = \frac{\kappa_1 + \kappa_2}{2} = \frac{1}{2\pi} \int_0^{2\pi} \kappa_n(\phi) \, d\phi
  \]

- **Gaussian curvature**
  \[
  K = \kappa_1 \cdot \kappa_2
  \]
Classification
Local surface shape by curvatures

Isotropic:
all directions are principal directions

Anisotropic:
2 distinct principal directions

\[ K > 0, \kappa_1 = \kappa_2 \]

spherical (umbilical)

\[ K = 0 \]
planar

\[ K > 0 \]
\[ \kappa_2 > 0, \kappa_1 > 0 \]
elliptic

\[ K = 0 \]
\[ \kappa_1 > 0, \kappa_2 = 0 \]
parabolic

\[ K < 0 \]
\[ \kappa_1 > 0, \kappa_2 < 0 \]
hyperbolic
Discrete Gaussian Curvature

- Angle deficit

\[ K = 2\pi - \sum_{i=1}^{N(v)} \phi_i \]
Mean Curvature

\[ H = \frac{1}{2\pi} \int_0^{2\pi} \kappa(\phi) \, d\phi \]

Can define through the Laplace-Beltrami operator

\[ \Delta_M \mathbf{p} = -H \mathbf{n} \]
Discrete Mean Curvature

- Intuition:

\[
H = \frac{1}{2\pi} \int_0^{2\pi} \kappa(\varphi) \, d\varphi
\]

\[
\gamma'' \approx \frac{1}{t} \left( (v_i - v_{i-1}) - (v_{i+1} - v_i) \right) = -\frac{1}{t} \left( v_{i-1} + v_{i+1} - 2v_i \right)
\]

\[\kappa \mathbf{n} = \gamma''\]
Discrete Laplace-Beltrami

- Intuition for uniform discretization

\[
H = \frac{1}{2\pi} \int_0^{2\pi} \kappa(\phi) \, d\phi
\]

\[
v_{j1} + v_{j4} - 2v_i + v_{j2} + v_{j5} - 2v_i + v_{j3} + v_{j6} - 2v_i =
\]

\[
6L(v_i) = \sum_{k=1}^{6} v_{jk} - 6v_i \approx -6Hn
\]
Discrete Laplace-Beltrami

- Intuition for uniform discretization

\[ H = \frac{1}{2\pi} \int_0^{2\pi} \kappa(\varphi) \, d\varphi \]

\[
\begin{align*}
\mathbf{v}_{j1} + \mathbf{v}_{j4} - 2\mathbf{v}_i &= + \\
\mathbf{v}_{j2} + \mathbf{v}_{j5} - 2\mathbf{v}_i &= + \\
\mathbf{v}_{i3} + \mathbf{v}_{i6} - 2\mathbf{v}_i &= = \\
L(\mathbf{v}_i) &= \frac{1}{6} \left( \sum_{k=1}^{6} \mathbf{v}_{jk} - 6\mathbf{v}_i \right) \approx -H\mathbf{n}
\end{align*}
\]
Discrete Laplace-Beltrami

- Cotangent formula – to compensate for triangle shape irregularity

\[
L_c(v_i) = \frac{1}{2A(v_i)} \sum_{v_j \in N_1(v_i)} \left( \cot \alpha_{ij} + \cot \beta_{ij} \right) (v_j - v_i)
\]
Discrete Laplace-Beltrami

- When the edge lengths are equal, the uniform and the cotangent Laplacians coincide
Discrete Laplace-Beltrami

- When the edge lengths are equal, the uniform and the cotangent Laplacians coincide
Linear Surface-Based Deformation

Prof. Dr. Mario Botsch

Computer Graphics & Geometry Processing
Bielefeld University
Mesh Deformation

Global deformation with intuitive detail preservation
Mesh Deformation

Local & global deformations
Linear Surface-Based Deformation

- Shell-Based Deformation
- Multiresolution Deformation
- Differential Coordinates
• Mesh deformation by displacement function $d$
  – Interpolate prescribed constraints
  – Smooth, intuitive deformation

$\Rightarrow$ Physically-based principles

$$d(p_i) = d_i$$

$$d : S \rightarrow \mathbb{R}^3$$

$$p \leftrightarrow p + d(p)$$
Shell Deformation Energy

• Stretching
  – Change of local distances
  – Captured by 1\textsuperscript{st} fundamental form

• Bending
  – Change of local curvature
  – Captured by 2\textsuperscript{nd} fundamental form

• Stretching & bending is sufficient
  – Differential geometry: “1\textsuperscript{st} and 2\textsuperscript{nd} fundamental forms determine a surface up to rigid motion.”
Physically-Based Deformation

- Nonlinear stretching & bending energies

\[ \int_{\Omega} k_s \| I - I' \|^2 + k_b \| \mathbf{II} - \mathbf{II}' \|^2 \, dudv \]

- Linearize terms $\rightarrow$ Quadratic energy

\[ \int_{\Omega} k_s \left( \| d_u \|^2 + \| d_v \|^2 \right) + k_b \left( \| d_{uu} \|^2 + 2 \| d_{uv} \|^2 + \| d_{vv} \|^2 \right) \, dudv \]
Physically-Based Deformation

• Minimize linearized bending energy

\[ E(d) = \int_S \|d_{uu}\|^2 + 2\|d_{uv}\|^2 + \|d_{vv}\|^2 \, dudv \rightarrow \min \]

\[ f(x) \rightarrow \min \]

• Variational calculus \(\rightarrow\) Euler-Lagrange PDE

\[ \Delta^2 d := d_{uuuu} + 2d_{uuvv} + d_{vvvv} = 0 \]

\[ f'(x) = 0 \]

\(\Rightarrow\) “Best” deformation that satisfies constraints
Deformation Energies

Initial state

$\Delta d = 0$
(Membrane)

$\Delta^2 d = 0$
(Thin plate)
PDE Discretization

- Euler-Lagrange PDE

\[ \Delta^2 d = 0 \]
\[ d = 0 \]
\[ d = \delta h \]

- Laplace discretization

\[
\Delta d_i = \frac{1}{2A_i} \sum_{j \in \mathcal{N}_i} (\cot \alpha_{ij} + \cot \beta_{ij})(d_j - d_i)
\]

\[
\Delta^2 d_i = \Delta (\Delta d_i)
\]
Linear System

- Sparse linear system (19 nz/row)

\[
\begin{pmatrix}
\Delta^2 \\
0 & I & 0 \\
0 & 0 & I
\end{pmatrix}
\begin{pmatrix}
d_i \\
\vdots
\end{pmatrix}
= 
\begin{pmatrix}
0 \\
0 \\
\delta h_i
\end{pmatrix}
\]

- Turn into symmetric positive definite system

- Solve this system *each frame*
  - Use efficient linear solvers !!!
  - Sparse Cholesky factorization
  - See course notes for details
Derivation Steps

1. Nonlinear Energy
2. Linearization
3. Quadratic Energy
4. Variational Calculus
5. Linear PDE
6. Discretization
7. Linear Equations
CAD-Like Deformation

[Botsch & Kobbelt, SIGGRAPH 04]
Face Animation

[Bickel et al, SCA 08]
Linear Surface-Based Deformation

- Shell-Based Deformation
- Multiresolution Deformation
- Differential Coordinates
Even pure translations induce local rotations!
⇒ Inherently non-linear coupling

Alternative approach
– Linear deformation + multi-scale decomposition...
Multiresolution Editing

Frequency decomposition

Change low frequencies

Add high frequency details, stored in local frames
Multiresolution Editing

Decomposition → Detail Information

Multiresolution Modeling

Freeform Modeling → Reconstruction

$S$ → $B$ → $B'$ → $S'$
Normal Displacements
Limitations

• Neighboring displacements are not coupled
  – Surface bending changes their angle
  – Leads to volume changes or self-intersections
Limitations

- Neighboring displacements are not coupled
  - Surface bending changes their angle
  - Leads to volume changes or self-intersections
Limitations

• Neighboring displacements are not coupled
  – Surface bending changes their angle
  – Leads to volume changes or self-intersections
  – See course notes for some other techniques...

• Multiresolution hierarchy difficult to compute
  – Complex topology
  – Complex geometry
  – Might require more hierarchy levels
Linear Surface-Based Deformation

- Shell-Based Deformation
- Multiresolution Deformation
- Differential Coordinates
1. Manipulate *differential coordinates* instead of *spatial* coordinates
   - Gradients, Laplacians, local frames
   - Intuition: Close connection to surface normal

2. Find mesh with desired differential coords
   - Cannot be solved exactly
   - Formulate as energy minimization
Differential Coordinates

Original → Rotated Diff-Coords → Reconstructed Mesh
Differential Coordinates

• Which differential coordinate $\delta_i$?
  – Gradients
  – Laplacians
  – ...

• How to get local transformations $T_i(\delta_i)$?
  – Smooth propagation
  – Implicit optimization
  – ...

Gradient-Based Editing

• Manipulate gradient of a function (e.g. a surface)

\[ g = \nabla f \quad g \mapsto T(g) \]

• Find function \( f' \) whose gradient is (close to) \( g' \)

\[ f' = \arg\min_f \int_\Omega \| \nabla f - T(g) \|^2 \, du \, dv \]

• Variational calculus \( \rightarrow \) Euler-Lagrange PDE

\[ \Delta f' = \text{div} \, T(g) \]
Gradient-Based Editing

- Consider piecewise linear *coordinate function*

\[ p(u, v) = \sum_{v_i} p_i \cdot \phi_i(u, v) \]

- Its gradient is

\[ \nabla p(u, v) = \sum_{v_i} p_i \cdot \nabla \phi_i(u, v) \]
Gradient-Based Editing

- Consider piecewise linear *coordinate function*

  \[ p(u, v) = \sum_{i} p_i \cdot \phi_i(u, v) \]

- Its gradient is

  \[ \nabla p(u, v) = \sum_{i} p_i \cdot \nabla \phi_i(u, v) \]

- It is constant per triangle

  \[ \nabla p|_{f_j} =: g_j \in \mathbb{R}^{3 \times 3} \]
Gradient-Based Editing

- Gradient of coordinate function \( p \)

\[
\begin{pmatrix}
g_1 \\
\vdots \\
g_F \\
\end{pmatrix}
= \mathbf{G} \\
\begin{pmatrix}
p_1^T \\
\vdots \\
p_V^T \\
\end{pmatrix}
\]

- Manipulate per-face gradients

\( g_j \mapsto T_j(g_j) \)
Gradient-Based Editing

• Reconstruct mesh from new gradients
  – Overdetermined \((3F \times V)\) system
  – Weighted least squares system
  ➡ Linear Poisson system \(\Delta p' = \text{div } T(g)\)

\[
\begin{align*}
\text{div} \nabla &= \Delta \\
G^T DG \cdot \begin{pmatrix} p'_1^T \\ \vdots \\ p'_V^T \end{pmatrix} &= G^T D \cdot \begin{pmatrix} T_1(g_1) \\ \vdots \\ T_F(g_F) \end{pmatrix}
\end{align*}
\]
Laplacian-Based Editing

• Manipulate Laplacians field of a surface
  \[ l = \Delta(p), \quad l \mapsto T(l) \]

• Find surface whose Laplacian is (close to) \( \delta' \)
  \[
p' = \arg\min_p \int_{\Omega} \| \Delta p - T(l) \|^2 \, dudv
  \]

• Variational calculus yields Euler-Lagrange PDE
  \[ \Delta^2 p' = \Delta T(l) \]
Differential Coordinates

• Which differential coordinate $\delta_i$ ?
  – Gradients
  – Laplacians
  – ...

• How to get local transformations $T_i(\delta_i)$ ?
  – Smooth propagation
  – Implicit optimization
  – ...
Smooth Propagation

1. Compute handle’s deformation gradient
2. Extract rotation and scale/shear components
3. Propagate damped rotations over ROI
Deformation Gradient

- Handle has been transformed *affinely*
  \[ T(x) = Ax + t \]

- Deformation gradient is
  \[ \nabla T(x) = A \]

- Extract rotation \( R \) and scale/shear \( S \)
  \[ A = U\Sigma V^T \quad \Rightarrow \quad R = UV^T, \quad S = V\Sigma V^T \]
Smooth Propagation

- Construct smooth scalar field $[0,1]$
  - $s(x)=1$: Full deformation (handle)
  - $s(x)=0$: No deformation (fixed part)
  - $s(x)\in(0,1)$: Damp handle transformation (in between)
Limitations

• Differential coordinates work well for rotations
  – Represented by deformation gradient

• Translations don’t change deformation gradient
  – Translations don’t change differential coordinates
  – “Translation insensitivity”
Implicit Optimization

- Optimize for positions $p_i'$ & transformations $T_i$

\[ \Delta^2 \begin{pmatrix} \vdots \\ p_i' \end{pmatrix} = \begin{pmatrix} \vdots \\ \Delta T_i(l_i) \end{pmatrix} \quad \leftrightarrow \quad T_i(p_i - p_j) = p_i' - p_j \]

- Linearize rotation/scale $\rightarrow$ one linear system

\[ T_i = \begin{pmatrix} s & -r_3 & r_2 \\ r_3 & s & -r_1 \\ -r_2 & r_1 & s \end{pmatrix} \]
Connection to Shells?

• Neglect local transformations $T_i$ for a moment...

$$\int \| \Delta p' - l \|^2 \rightarrow \min$$

$$\Delta^2 p' = \Delta l$$

- Basic formulations equivalent!
- Differ in detail preservation
  - Rotation of Laplacians
  - Multi-scale decomposition

$$\int \| d_{uu} \|^2 + 2 \| d_{uv} \|^2 + \| d_{vv} \|^2 \rightarrow \min$$

$$\Delta^2 d = 0$$

$$\Delta^2 (p + d) = \Delta^2 p$$

$$p' = p + d$$

$$l = \Delta p$$
Linear Surface-Based Deformation

- Shell-Based Deformation
- Multiresolution Deformation
- Differential Coordinates
Interactive Shape Modeling and Deformation

T3: Half-Day Tutorial

Linear Space Deformations
Space Deformation

- Displacement function defined on the ambient space
  \[ d : \mathbb{R}^3 \rightarrow \mathbb{R}^3 \]

- Evaluate the function on the points of the shape embedded in the space
  \[ x' = x + d(x) \]

Twist warp
Global and local deformation of solids
[A. Barr, SIGGRAPH 84]
Freeform Deformations

- Control object
- User defines displacements $d_i$ for each element of the control object
- Displacements are interpolated to the entire space using basis functions $B_i(x) : \mathbb{R}^3 \rightarrow \mathbb{R}$

\[
d(x) = \sum_{i=1}^{k} d_i B_i(x)
\]

- Basis functions should be smooth for aesthetic results
Trivariate Tensor Product Bases

[Sederberg and Parry 86]

- Control object = lattice
- Basis functions $B_i(x)$ are trivariate tensor-product splines:

$$d(x, y, z) = \sum_{i=0}^{l} \sum_{j=0}^{m} \sum_{k=0}^{n} d_{ijk} N_i(x) N_j(y) N_i(z)$$
Trivariate Tensor Product Bases

- Similar to the surface case
  - Aliasing artifacts

- Interpolate deformation constraints?
  - Only in least squares sense
Lattice as Control Object

- Difficult to manipulate
- The control object is not related to the shape of the edited object
- Part of the shape in close Euclidean distance always deform similarly, even if geodesically far
Wires

[Singh and Fiume 98]

- Control objects are arbitrary space curves
- Can place curves along meaningful features of the edited object
- Smooth deformations around the curve with decreasing influence
Handle Metaphor
[RBF, Botsch and Kobbelt 05]

- Wish list for the displacement function $d(x)$:
  - Interpolate prescribed constraints
  - Smooth, intuitive deformation
Volumetric Energy Minimization

[RBF, Botsch and Kobbelt 05]

- Minimize similar energies to surface case
  \[
  \int_{\mathbb{R}^3} \left( \|d_{xx}\|^2 + \|d_{xy}\|^2 + \ldots + \|d_{zz}\|^2 \right) \, dx dy dz \rightarrow \min
  \]

- But displacements function lives in 3D...
  - Need a volumetric space tessellation?
  - No, same functionality provided by RBFs!
Radial Basis Functions

[RBF, Botsch and Kobbelt 05]

- Represent deformation by RBFs
  \[ d(x) = \sum_j w_j \cdot \varphi(|| c_j - x ||) + p(x) \]

- Triharmonic basis function \( \varphi(r) = r^3 \)
  - \( C^2 \) boundary constraints
  - Highly smooth / fair interpolation

\[ \int_{\mathbb{R}^3} \left( \sum_k \left\| d_{xxx} \right\|^2 + \left\| d_{xyy} \right\|^2 + \ldots + \left\| d_{zzz} \right\|^2 \right) \, dx \, dy \, dz \rightarrow \min \]
RBF Fitting

[RBF, Botsch and Kobbelt 05]

- Represent deformation by RBFs

\[ d(x) = \sum_j w_j \cdot \varphi(\|c_j - x\|) + p(x) \]

- RBF fitting
  - Interpolate displacement constraints
  - Solve linear system for \( w_j \) and \( p \)
RBF Fitting

[RBF, Botsch and Kobbelt 05]

- Represent deformation by RBFs

\[ d(x) = \sum_j w_j \cdot \varphi(||c_j - x||) + p(x) \]

- RBF evaluation
  - Function \( d \) transforms points
  - Jacobian \( \nabla d \) transforms normals
  - Precompute basis functions
  - Evaluate on the GPU!
Local & Global Deformations

[RBF, Botsch and Kobbelt 05]
Local & Global Deformations

[RBF, Botsch and Kobbelt 05]

1M vertices
movie
Space Deformations

Summary so far

- Handle arbitrary input
  - Meshes (also non-manifold)
  - Point sets
  - Polygonal soups
  - ...

- Complexity mainly depends on the control object, not the surface

- 3M triangles
- 10k components
- Not oriented
- Not manifold
Space Deformations
Summary so far

- Handle arbitrary input
  - Meshes (also non-manifold)
  - Point sets
  - Polygonal soups
  - ...

- Easier to analyze: functions on Euclidean domain
  - Volume preservation: $|\text{Jacobian}| = 1$
Space Deformations

Summary so far

- The deformation is only loosely aware of the shape that is being edited
- Small Euclidean distance → similar deformation
- Local surface detail may be distorted
Cage-based Deformations

[Ju et al. 2005]

- Cage = crude version of the input shape
- Polytope (not a lattice)
Cage-based Deformations

[Ju et al. 2005]

- Cage = crude version of the input shape
- Polytope (not a lattice)
- Each point $\mathbf{x}$ in space is represented w.r.t. to the cage elements using coordinate functions

\[
\mathbf{x} = \sum_{i=1}^{k} w_i(\mathbf{x}) \mathbf{p}_i
\]
Cage-based Deformations

[Cu et al. 2005]

- Cage = crude version of the input shape
- Polytope (not a lattice)
- Each point $x$ in space is represented w.r.t. to the cage elements using coordinate functions

\[ x = \sum_{i=1}^{k} w_i(x) p_i \]
Cage-based Deformations

[Ju et al. 2005]

- Cage = crude version of the input shape
- Polytope (not a lattice)
Cage-based Deformations

[Ju et al. 2005]

- Cage = crude version of the input shape
- Polytope (not a lattice)

\[ x' = \sum_{i=1}^{k} w_i(x) p'_i \]
Cage-based Deformations

- Cage = crude version of the input shape
- Polytope (not a lattice)

\[ x' = \sum_{i=1}^{k} w_i(x)p'_i \]
Coordinate Functions

- Mean-value coordinates (Floater, Ju et al. 2005)
  - Generalization of barycentric coordinates
  - Closed-form solution for $w_i(x)$
Coordinate Functions

- Mean-value coordinates (Floater, Ju et al. 2005)
  - Not necessarily positive on non-convex domains

MVC
Coordinate Functions

- PMVC (Lipman et al. 2007) – ensures positivity, but no longer closed-form and only $C^0$
Coordinate Functions

- Harmonic coordinates (Joshi et al. 2007)
  - Harmonic functions $h_i(x)$ for each cage vertex $p_i$
  - Solve
    \[ \Delta h = 0 \]
    subject to: $h_i$ linear on the boundary s.t. $h_i(p_i) = \delta_{ij}$
Coordinate Functions

- Harmonic coordinates (Joshi et al. 2007)
  - Harmonic functions $h_i(x)$ for each cage vertex $p_i$
  - Solve
    \[ \Delta h = 0 \]
    subject to: $h_i$ linear on the boundary s.t. $h_i(p_i) = \delta_{ij}$

- Volumetric Laplace equation
- Discretization, no closed-form
Coordinate Functions

- Harmonic coordinates (Joshi et al. 2007)
Coordinate Functions

- Green coordinates (Lipman et al. 2008)
- Observation: previous vertex-based basis functions always lead to affine-invariance!

\[ x' = \sum_{i=1}^{k} w_i(x)p_i' \]
Coordinate Functions

- Green coordinates (Lipman et al. 2008)
- Correction: Make the coordinates depend on the cage faces as well

\[ x' = \sum_{i=1}^{k} w_i(x)p'_i + \sum_{j=1}^{m} \psi_j(x)n'_j \]
Coordinate Functions

- Green coordinates (Lipman et al. 2008)
- Closed-form solution
- Conformal in 2D, quasi-conformal in 3D
Coordinate Functions

- Green coordinates (Lipman et al. 2008)
- Closed-form solution
- Conformal in 2D, quasi-conformal in 3D

Alternative interpretation in 2D via holomorphic functions and extension to point handles: Weber et al. Eurographics 2009
Coffee/Tea Break

Resume at 11:00
Summary of Linear Methods

Prof. Dr. Mario Botsch

Computer Graphics & Geometry Processing
Bielefeld University
Linear Approaches

Nonlinear Energy

$\text{Linearization}$

Quadratic Energy

$\text{Variational Calculus}$

Linear PDE

$\text{Discretization}$

Linear Equations
Linear Approaches

• Resulting linear systems
  – Shell-based $\Delta^2 d = 0$
  – Gradient-based $\Delta p = \nabla \cdot T(g)$
  – Laplacian-based $\Delta^2 p = \Delta T(l)$

• Properties
  – Highly sparse
  – Symmetric, positive definite \((SPD)\)
  – Solve for new RHS each frame!
Linear SPD Solvers

• Dense Cholesky factorization
  – Cubic complexity
  – High memory consumption (doesn’t exploit sparsity)

• Iterative conjugate gradients
  – Quadratic complexity
  – Need sophisticated preconditioning

• Multigrid solvers
  – Linear complexity
  – But rather complicated to develop (and to use)

• Sparse Cholesky factorization
  – Linear complexity
  – Easy to use
Dense Cholesky Factorization

Solve \( Ax = b \)

1. Cholesky factorization \( A = LL^T \)

2. Solve system \( y = L^{-1}b, \quad x = L^{-T}y \)
Dense Cholesky Factorization

\[ A = LL^T \]

500×500 matrix
3500 non-zeros

Cholesky Factorization

\[ L \]

36k non-zeros
Sparse Cholesky Factorization

\[ A = LL^T \]

500×500 matrix
3500 non-zeros

Reordering

\[ P^T A P \]

Cholesky Factorization

\[ L \]

36k non-zeros

14k non-zeros
Sparse Cholesky Factorization

\[ A = LL^T \]

500x500 matrix
3500 non-zeros

Reordering

\[ \text{PTAP} \]

Cholesky Factorization

\[ L \]

36k non-zeros

7k non-zeros
Sparse Cholesky Solver

Solve $Ax = b$

Pre-computation

1. Matrix re-ordering $\tilde{A} = P^TAP$
2. Cholesky factorization $\tilde{A} = LL^T$
3. Solve system $y = L^{-1}P^Tb$, $x = PL^{-T}y$

Per-frame computation
Bi-Laplace Systems

3 Solutions (per frame costs)

- Conjugate Gradients
- Multigrid
- Sparse Cholesky
Linear Approaches

Nonlinear Energy

Linearization

Quadratic Energy

Variational Calculus

Linear PDE

Discretization

Linear Equations
Linear vs. Nonlinear

Shell

Gradient

Nonlinear
Linear Approaches

Nonlinear Energy

\[ \text{Linearization} \]

Quadratic Energy

\[ \text{Variational Calculus} \]

Linear PDE

\[ \text{Discretization} \]

Linear Equations

causes artifacts
• **Shell-based deformation**

\[
\int_{\Omega} k_s \| \mathbf{I} - \mathbf{I}' \|^2 + k_b \| \mathbf{II} - \mathbf{II}' \|^2 \, dudv
\]

\[
\int_{\Omega} k_s \left( \| \mathbf{d}_u \|^2 + \| \mathbf{d}_v \|^2 \right) + k_b \left( \| \mathbf{d}_{uu} \|^2 + 2 \| \mathbf{d}_{uv} \|^2 + \| \mathbf{d}_{vv} \|^2 \right) \, dudv
\]
• Gradient-based editing

$$\nabla T(x) = A$$
• Laplacian surface editing

\[ R x \approx x + (r \times x) = \begin{pmatrix} 1 & -r_3 & r_2 \\ r_3 & 1 & -r_1 \\ -r_2 & r_1 & 1 \end{pmatrix} x \]

\[ T_i = \begin{pmatrix} s & -r_3 & r_2 \\ r_3 & s & -r_1 \\ -r_2 & r_1 & s \end{pmatrix} \]
Linear vs. Nonlinear

- Analyze existing methods
  - Some work for translations
  - Some work for rotations
  - No method works for both

<table>
<thead>
<tr>
<th>Nonlinear</th>
<th>Linear</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shell</td>
<td></td>
</tr>
<tr>
<td>Gradient</td>
<td></td>
</tr>
<tr>
<td>Laplace</td>
<td></td>
</tr>
</tbody>
</table>
Linear vs. Nonlinear

• Linear approaches
  – Solve linear system each frame
  – Small deformations
  – Dense constraints

• Nonlinear approaches
  – Solve nonlinear problem each frame
  – Large deformations
  – Sparse constraints
Nonlinear Surface-Based Deformation

Prof. Dr. Mario Botsch
Computer Graphics & Geometry Processing
Bielefeld University
Nonlinear Surface Deformation

- Nonlinear Optimization
- Shell-Based Deformation
- (Differential Coordinates)
Nonlinear Minimization

• Given a nonlinear deformation energy

\[ E(d) = E(d_1, \ldots, d_n) \]

find the displacement \( d(x) \) that minimizes \( E(d) \), while satisfying the modeling constraints.

• Typically \( E(d) \) stays the same, but the modeling constraints change each frame.
Gradient Descent

- Start with initial guess $d_0$
- Iterate until convergence
  - Find descent direction $h = -\nabla E(d)$
  - Find step size $\lambda$
  - Update $d = d + \lambda h$

- Properties
  + Easy to implement, guaranteed convergence
  - Slow convergence
Newton’s Method

• Start with initial guess \( d_0 \)

• Iterate until convergence
  – Find descent direction as \( H(d) \ h = -\nabla E(d) \)
  – Find step size \( \lambda \)
  – Update \( d = d + \lambda h \)

• Properties
  + Fast convergence if close to minimum
  – Needs pos. def. \( H \), needs 2\(^{nd}\) derivatives for \( H \)
Nonlinear Least Squares

Given a nonlinear vector-valued error function

\[ e(d_1, \ldots, d_n) = \begin{pmatrix} e_1(d_1, \ldots, d_n) \\ \vdots \\ e_m(d_1, \ldots, d_n) \end{pmatrix} \]

find the displacement \( d(x) \) that minimizes the nonlinear least squares error

\[ E(d_1, \ldots, d_n) = \frac{1}{2} \| e(d_1, \ldots, d_n) \|^2 \]
Gauss-Newton Method

• Start with initial guess \( d_0 \)

• Iterate until convergence
  – Find descent direction as \( (J(d)^T J(d)) \ h = -J(d)^T e \)
  – Find step size \( \lambda \)
  – Update \( d = d + \lambda h \)

• Properties
  + Fast convergence if close to minimum
  + Needs full-rank \( J(d) \), needs 1\(^{st}\) derivatives for \( J(d) \)
Nonlinear Optimization

• Has to solve a linear system each frame
  – Matrix changes in each iteration!
  – Factorize matrix each time

• Numerically more complex
  – No guaranteed convergence
  – Might need several iterations
  – Converges to closest local minimum

➡ Spend more time on fancy solvers...
Nonlinear Surface Deformation

- Nonlinear Optimization
- Shell-Based Deformation
- (Differential Coordinates)
Shell-Based Deformation

- Discrete Shells
  [Grinspun et al, SCA 2003]

- Rigid Cells
  [Botsch et al, SGP 2006]

- As-Rigid-As-Possible Modeling
  [Sorkine & Alexa, SGP 2007]
Discrete Shells

• Main idea
  – Don’t discretize continuous energy
  – Define **discrete** energy instead
  – Leads to simpler (still nonlinear) formulation

• Discrete energy
  – How to measure stretching on meshes?
  – How to measure bending on meshes?
Discrete Shell Energy

- **Stretching**: Change of edge lengths
  \[
  \sum_{e_{ij} \in E} \lambda_{ij} \left( |e_{ij}| - |\bar{e}_{ij}| \right)^2
  \]

- **Stretching**: Change of triangle areas
  \[
  \sum_{f_{ijk} \in F} \lambda_{ijk} \left( |f_{ijk}| - |\bar{f}_{ijk}| \right)^2
  \]

- **Bending**: Change of dihedral angles
  \[
  \sum_{e_{ij} \in E} \mu_{ij} \left( \theta_{ij} - \bar{\theta}_{ij} \right)^2
  \]
Discrete Shells

[Grinspun 2003]
Realistic Face Animations

Linear model

Nonlinear model
Discrete Energy Gradients

- Gradients of edge length

\[
|e_{ij}| = \|x_j - x_i\|
\]

\[
\frac{\partial |e_{ij}|}{\partial x_i} = -\frac{e}{\|e\|}
\]

\[
\frac{\partial |e_{ij}|}{\partial x_j} = \frac{e}{\|e\|}
\]
Discrete Energy Gradients

- Gradients of triangle area

\[ \begin{align*}
|f_{ijk}| & = \frac{1}{2} \|n_1\| \\
\frac{\partial |f_{ijk}|}{\partial x_i} & = \frac{n_1 \times (x_k - x_j)}{2 \|n_1\|} \\
\frac{\partial |f_{ijk}|}{\partial x_j} & = \frac{n_1 \times (x_i - x_k)}{2 \|n_1\|} \\
\frac{\partial |f_{ijk}|}{\partial x_k} & = \frac{n_1 \times (x_j - x_i)}{2 \|n_1\|}
\end{align*} \]
Discrete Energy Gradients

• Gradients of dihedral angle

\[
\theta = \arctan\left(\frac{\sin \theta}{\cos \theta}\right) = \arctan\left(\frac{(n_1 \times n_2)^T e}{n_1^T n_2 \cdot \|e\|}\right)
\]

\[
\frac{\partial \theta}{\partial x_i} = \frac{(x_k - x_j)^T e}{\|e\|} \cdot \frac{-n_1}{\|n_1\|^2} + \frac{(x_l - x_j)^T e}{\|e\|} \cdot \frac{-n_2}{\|n_2\|^2}
\]

\[
\frac{\partial \theta}{\partial x_j} = \frac{(x_i - x_k)^T e}{\|e\|} \cdot \frac{-n_1}{\|n_1\|^2} + \frac{(x_i - x_l)^T e}{\|e\|} \cdot \frac{-n_2}{\|n_2\|^2}
\]

\[
\frac{\partial \theta}{\partial x_k} = \|e\| \cdot \frac{-n_1}{\|n_1\|^2}
\]

\[
\frac{\partial \theta}{\partial x_l} = \|e\| \cdot \frac{-n_2}{\|n_2\|^2}
\]
Discrete Shell Editing

• Problems with large deformation
  – Bad initial state causes numerical problems
Shell-Based Deformation

• Discrete Shells
  [Grinspun et al, SCA 2003]

• Rigid Cells
  [Botsch et al, SGP 2006]

• As-Rigid-As-Possible Modeling
  [Sorkine & Alexa, SGP 2007]
Nonlinear Shape Deformation

- *Nonlinear* editing too instable?

- *Physically plausible* vs. physically correct

  - Trade physical correctness for
    - Computational efficiency
    - Numerical robustness
• Qualitatively emulate thin-shell behavior
• Thin volumetric layer around center surface
• Extrude polygonal cell $C_i$ per mesh face
• Aim for robustness
  – Prevent cells from degenerating
  ➡ Keep cells *rigid*
Elastically Connected Rigid Cells

- Connect cells along their faces
  - Nonlinear elastic energy
  - Measures bending, stretching, twisting, ...
Cell-Based Surface Deformation

1. Prescribes position/orientation for cells
2. Find optimal rigid motions per cell
3. Update vertices by average cell transformations
Elastically Connected Rigid Cells

- **Pairwise energy**

\[ E_{ij} = \int_{[0,1]^2} \| f^{i\rightarrow j}(u) - f^{j\rightarrow i}(u) \|^2 \, du \]

- **Global energy**

\[ E = \sum_{\{i,j\}} w_{ij} \cdot E_{ij} \quad w_{ij} = \frac{\| e_{ij} \|^2}{|F_i| + |F_j|} \]
• Find rigid motion $T_i$ per cell $C_i$

$$\min_{\{T_i\}} \sum_{i,j} w_{ij} \int_{[0,1]^2} \| T_i(f^{i\rightarrow j}(u)) - T_j(f^{j\rightarrow i}(u)) \|^2 \, du$$

• Generalized global shape matching problem
  – Robust geometric optimization
  – Nonlinear Newton-type minimization
  – Hierarchical multi-grid solver
Newton-Type Iteration

1. Linearization of rigid motions

\[ R_i x + t_i \approx x + (\omega_i \times x) + v_i =: A_i x \]

2. Quadratic optimization of velocities

\[
\min_{\{v_i, \omega_i\}} \sum_{i,j} w_{ij} \int_{[0,1]^2} \| A_i(f^{i\to j}(u)) - A_j(f^{j\to i}(u)) \| ^2 \, du
\]

3. Project \( A_i \) onto rigid motion manifold

\[ \rightarrow \text{Local shape matching} \]
Robustness
Character Posing
Goblin Posing

- Intuitive large scale deformations
- Whole session < 5 min
Shell-Based Deformation

• Discrete Shells
  [Grinspun et al, SCA 2003]

• Rigid Cells
  [Botsch et al, SGP 2006]

• As-Rigid-As-Possible Modeling
  [Sorkine & Alexa, SGP 2007]
Surface Deformation

- Smooth large scale deformation
- Local as-rigid-as-possible behavior
  - Preserves small-scale details
Cell Deformation Energy

- Vertex neighborhoods should deform rigidly

\[ \sum_{j \in N(i)} \left\| (p'_j - p'_i) - R_i (p_j - p_i) \right\|^2 \rightarrow \min \]
Cell Deformation Energy

• If $p, p'$ are known then $R_i$ is uniquely defined

*Shape matching* problem
  – Build covariance matrix $S = PP'^T$
  – SVD: $S = U\Sigma W^T$
  – Extract rotation $R_i = UW^T$
Total Deformation Energy

• Sum over all vertex

$$\min_{p'} \sum_{i=1}^{n} \sum_{j \in N(i)} \| (p'_j - p'_i) - R_i (p_j - p_i) \|^2$$

• Treat $p'$ and $R_i$ as separate variables

• Allows for alternating optimization
  – Fix $p'$, find $R_i$ : Local shape matching per cell
  – Fix $R_i$, find $p'$ : Solve Laplacian system
As-Rigid-As-Possible Modeling

• Start from naïve Laplacian editing as initial guess

initial guess 1 iteration 2 iterations

initial guess 1 iterations 4 iterations
Shell-Based Deformation

- Discrete Shells
  [Grinspun et al, SCA 2003]

- Rigid Cells
  [Botsch et al, SGP 2006]

- As-Rigid-As-Possible Modeling
  [Sorkine & Alexa, SGP 2007]
Nonlinear Surface Deformation

• Limitations of Linear Methods
• Shell-Based Deformation
• (Differential Coordinates)
Subspace Gradient Deformation

- Nonlinear Laplacian coordinates
- Least squares solution on coarse cage subspace

[Huang et al, SIGGRAPH 06]
Mesh Puppetry

- Skeletons and Laplacian coordinates
- Cascading optimization

[Shi et al, SIGGRAPH 07]
Nonlinear Surface Deformation

- Limitations of Linear Methods
- Shell-Based Deformation
- (Differential Coordinates)
Interactive Shape Modeling and Deformation

T3: Half-Day Tutorial

Nonlinear Space Deformations
Nonlinear Space Deformations

- Involve nonlinear optimization
- Enjoy the advantages of space warps
- Additionally, have shape-preserving properties
As-Rigid-As-Possible Deformation
Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Points or segments as control objects
- First developed in 2D and later extended to 3D by Zhu and Gortler (2007)
As-Rigid-As-Possible Deformation
Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Attach an affine transformation to each point \( x \in \mathbb{R}^3 \):
  \[
  A_x(p) = M_x p + t_x
  \]

- The space warp:
  \( x \rightarrow A_x(x) \)
As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Handles $p_i$ are displaced to $q_i$
- The local transformation at $x$:
  \[ A_x(p) = M_x p + t_x \quad \text{s.t.} \]
  \[ \sum_{i=1}^{k} w_i(x) \| A_x(p_i) - q_i \|^2 \rightarrow \min \]
- The weights depend on $x$:
  \[ w_i(x) = \| p_i - x \|^{-2\alpha} \]
As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- No additional restriction on $A_x(\cdot)$ – affine local transformations
As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Restrict $A_x(\cdot)$ to similarity
As-Rigid-As-Possible Deformation
Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Restrict $A_x(\cdot)$ to similarity

\[
M_x = \begin{pmatrix}
a & b \\
-b & a
\end{pmatrix}
\]
As-Rigid-As-Possible Deformation
Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Restrict $A_x(\cdot)$ to rigid
As-Rigid-As-Possible Deformation
Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Restrict $A_x(\cdot)$ to rigid

$$M_x = \begin{pmatrix} a & b \\ -b & a \end{pmatrix}$$

Solve for $M_x$ like similarity and then normalize
As-Rigid-As-Possible Deformation
Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Examples
As-Rigid-As-Possible Deformation

MLS approach – extension to 3D [Zhu & Gortler 2007]

- No linear expression for similarity in 3D
- Instead, can solve for the minimizing rotation

\[
\arg\min_{R\in\text{SO}(3)} \sum_{i=1}^{k} w_i(x) \|Rp_i - q_i\|^2
\]

by polar decomposition of the $3\times3$ covariance matrix
As-Rigid-As-Possible Deformation

MLS approach – extension to 3D [Zhu & Gortler 2007]

- Zhu and Gortler also replace the Euclidean distance in the weights by “distance within the shape”

\[ w_i(x) = d(p_i, x)^{-2\alpha} \]
As-Rigid-As-Possible Deformation

MLS approach – extension to 3D [Zhu & Gortler 2007]

- More results
As-Rigid-As-Possible Deformation

Deformation Graph approach [Sumner et al. 2007]

- Surface handles as interface
- Underlying graph to represent the deformation; nodes store rigid transformations
- Decoupling of handles from def. representation
Deformation Graph

[Sumner et al. 2007]
Begin with an embedded object.

Deformation Graph
[Sumner et al. 2007]
Deformation Graph

[Sumner et al. 2007]

Begin with an embedded object.
Nodes selected via uniform sampling; located at $g_j$.
One rigid transformation for each node: $R_j, t_j$.
Each node deforms nearby space.
Edges connect nodes of overlapping influence.
One rigid transformation for each node: $R_j, t_j$

Each node deforms nearby space.

Edges connect nodes of overlapping influence.

Begin with an embedded object.

Nodes selected via uniform sampling; located at $g_j$.

Deformation Graph [Sumner et al. 2007]
Influence of nearby transformations is blended.

$$x' = \sum_{j=1}^{m} w_j(x) \left[ R_j(x - g_j) + g_j + t_j \right]$$

$$w_j(x) = (1 - \|x - g_j\| / d_{\text{max}})^2$$
Select & drag vertices of embedded object.

Optimization
[Sumner et al. 2007]
Select & drag vertices of embedded object.

Optimization finds deformation parameters $R_j$, $t_j$. 

[Sumner et al. 2007]
$$\min_{R_1, t_1, \ldots, R_m, t_m} \quad w_{\text{rot}} E_{\text{rot}} + w_{\text{reg}} E_{\text{reg}} + w_{\text{con}} E_{\text{con}}$$

- **Graph parameters**
- **Rotation term**
- **Regularization term**
- **Constraint term**

Select & drag vertices of embedded object.

Optimization finds deformation parameters $R_j, t_j$.  

3/30/2009
\[
\min_{\mathbf{R}_1, \mathbf{t}_1, \ldots, \mathbf{R}_m, \mathbf{t}_m} \ w_{\text{rot}} \mathbf{E}_{\text{rot}} + w_{\text{reg}} \mathbf{E}_{\text{reg}} + w_{\text{con}} \mathbf{E}_{\text{con}}
\]

\[
\text{Rot}(\mathbf{R}) = (\mathbf{c}_1 \cdot \mathbf{c}_2)^2 + (\mathbf{c}_1 \cdot \mathbf{c}_3)^2 + (\mathbf{c}_2 \cdot \mathbf{c}_3)^2 + (\mathbf{c}_1 \cdot \mathbf{c}_1 - 1)^2 + (\mathbf{c}_2 \cdot \mathbf{c}_2 - 1)^2 + (\mathbf{c}_3 \cdot \mathbf{c}_3 - 1)^2
\]

\[
\mathbf{E}_{\text{rot}} = \sum_{j=1}^{m} \text{Rot}(\mathbf{R}_j)
\]

For detail preservation, features should rotate and not scale or skew.
$\min_{R_1, t_1, \ldots, R_m, t_m} \ w_{\text{rot}} E_{\text{rot}} + w_{\text{reg}} E_{\text{reg}} + w_{\text{con}} E_{\text{con}}$

$E_{\text{reg}} = \sum_{j=1}^{m} \sum_{k \in N(j)} \alpha_{jk} \left\| R_j(g_k - g_j) + g_j + t_j - (g_k + t_k) \right\|_2^2$

where node $j$ thinks node $k$ should go

where node $k$ actually goes

Neighboring nodes should agree on where they transform each other.
\[
\text{min}_{R_1, t_1, \ldots, R_m, t_m} \ w_{\text{rot}} E_{\text{rot}} + w_{\text{reg}} E_{\text{reg}} + w_{\text{con}} E_{\text{con}}
\]

\[
E_{\text{con}} = \sum_{l=1}^{p} \left\| \tilde{v}_{\text{index}(l)} - q_l \right\|_2^2
\]

Handle vertices should go where the user puts them.
\[
\min_{R_1, t_1, \ldots, R_m, t_m} \quad \mathcal{W}_{\text{rot}} E_{\text{rot}} + \mathcal{W}_{\text{reg}} E_{\text{reg}} + \mathcal{W}_{\text{con}} E_{\text{con}}
\]
Results: Polygon Soup

[Sumner et al. 2007]
Results: Giant Mesh

[Sumner et al. 2007]
Results: Detail Preservation

[Sumner et al. 2007]
Discussion

- Decoupling of deformation complexity and model complexity
- Nonlinear energy optimization – results comparable to surface-based approaches
Interactive Shape Modeling and Deformation

T3: Half-Day Tutorial

Wrap-up
Research trends

- From linear to nonlinear techniques
- Surface-based methods and space warps developed simultaneously
Future work?

- Higher-level editing
  - ... with semantic understanding of the shape
  - ... with “pseudo-physics” automatically set up from that understanding
- Hybrids between surface- and space-based methods
Acknowledgments

- The authors of the numerous papers presented in our tutorial, for their inspiring research
- Pierre Alliez, Eitan Grinspun, Tao Ju, Andrew Nealen, Mark Pauly, Scott Schaefer, Bob Sumner, Jens Vorsatz for kindly providing their slides

- Olga Sorkine’s presentation was supported by an ADVANCE Women-in-Science Travel Grant funded by the NSF ADVANCE PAID award #HRD-0820202